AI 응용시스템의 이해와 구축

9강. Large model + Knowledge distillation

출석.

Announcement

특강: 5/21 “Machine Learning in Audio Engineering” 이정석 박사 @ 메타 (Facebook)

Project: 코멘트 드리고 있습니다.

Giant Models

Giant Neural Nets

Vision:

- (2014) ImageNet 우승: GoogleNet 에 **4 million parameter** (74% top-1 accuracy)
- (2017) Squeeze-and-excitation network, **145 mil parameter** (82.7% top-1 acc)

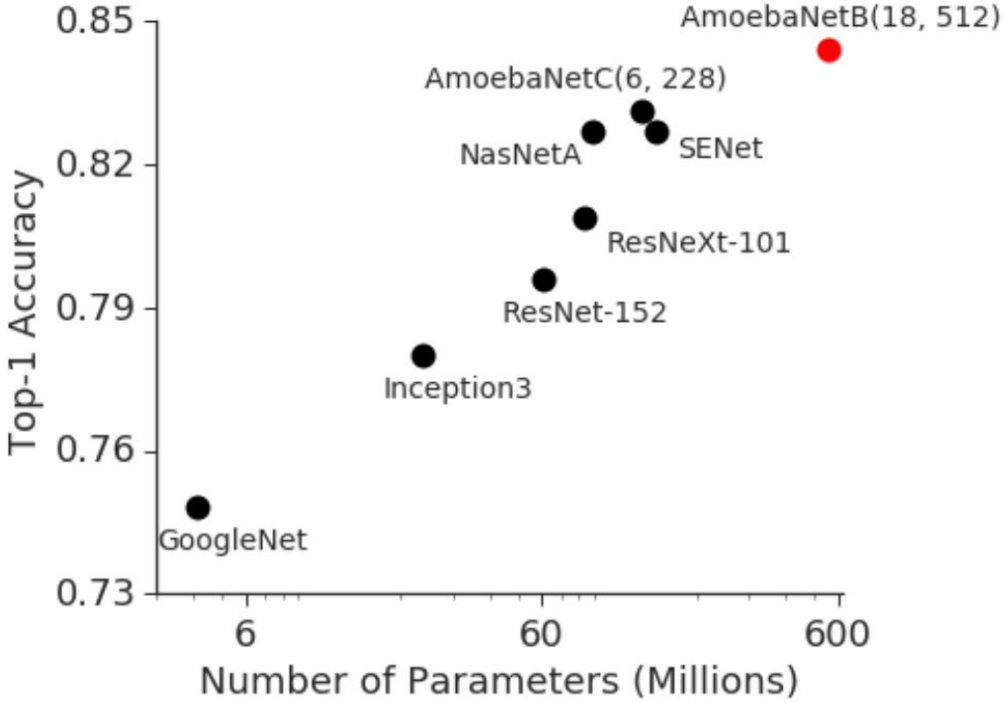
36 배의 parameter 증가.

Language:

- (2018)BERT: **110 million parameters** (base), 340 million (large)
- (2020) GPT3: **175 billion parameters**
- (2022) Pathways Language Model (PaLM): **540 billion parameter**

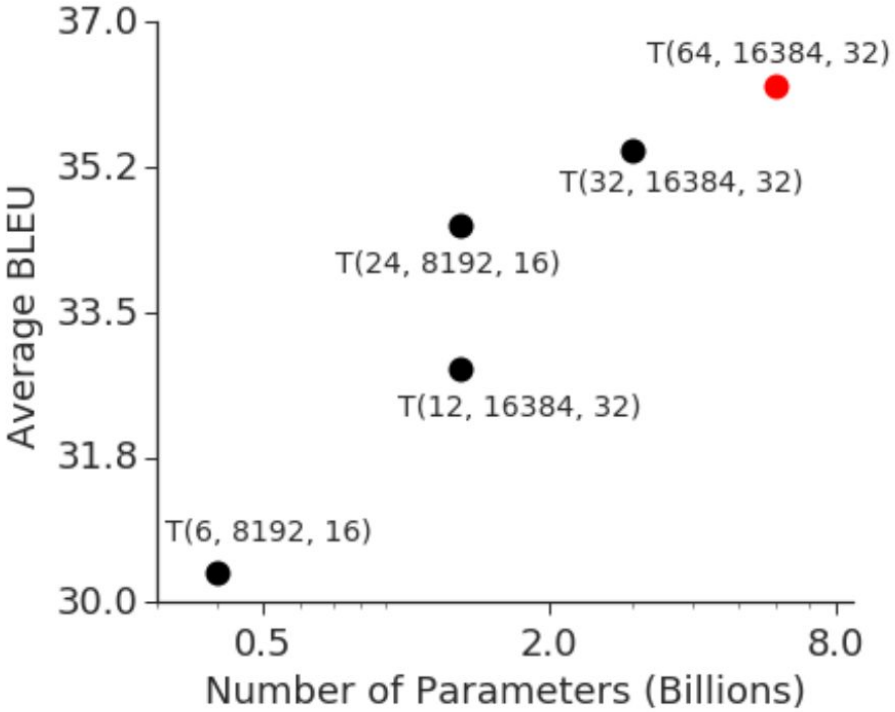
더이상 모델이 하나의 머신에서 학습할 수 없음..

Vision models



- (y-axis) Top-1 accuracy on ImageNet 2021
- (x-axis) model size
- AmoebaNetB: 84.4% with 550M params

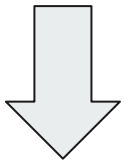
Language models



- (y-axis) Average improvement in translation quality (BLEU)
- (x-axis) model size
- T(L,H,A): transformer with
  - L: encoder /decoder layers
  - H: feed forward hidden dimension layer
  - A: attention layers
- 빨간 점: 128-layer **6Billion 파라메터** 트랜스포머.

Large network의 과제

- GPU 의 사이즈가 모델복잡도보다 더디게 발달.
- Cloud TPU에서도 메모리가 부족.



Large-scale training 테크놀로지가 필요.

Quantization을 통해 float32모델을 8비트, 4비트로 변환.

➔ 4~8x 사이즈 감소 (inference only)

메모리 부족을 해결하는 방법 (시스템레벨)

알고리즘레벨:
quantization, sparsity, distillation..

- Gradient Accumulation
  - Backpropagation 알고리즘을 통해 gradient을 계산할 때, mini-batch마다 계산해서 업데이트 하지 않고 gradient들을 축적 (accumulate). 전체 batch가 완성되면 축적된 gradient들을 backprop을 사용해 모델 업데이트 진행.
- Memory / CPU swap
  - GPU/TPU에 메모리가 부족하므로, activation을 CPU/RAM 으로 복사했다가 차후 다시 GPU/TPU로 복사.
  - Slow.

모델의 병렬처리

Model Parallelism:

- 모델을 k worker로 분할하여 학습진행.
- Forward pass / backward pass 할때마다 shared variables (activation)을 주고 받아야 함.

Naive:

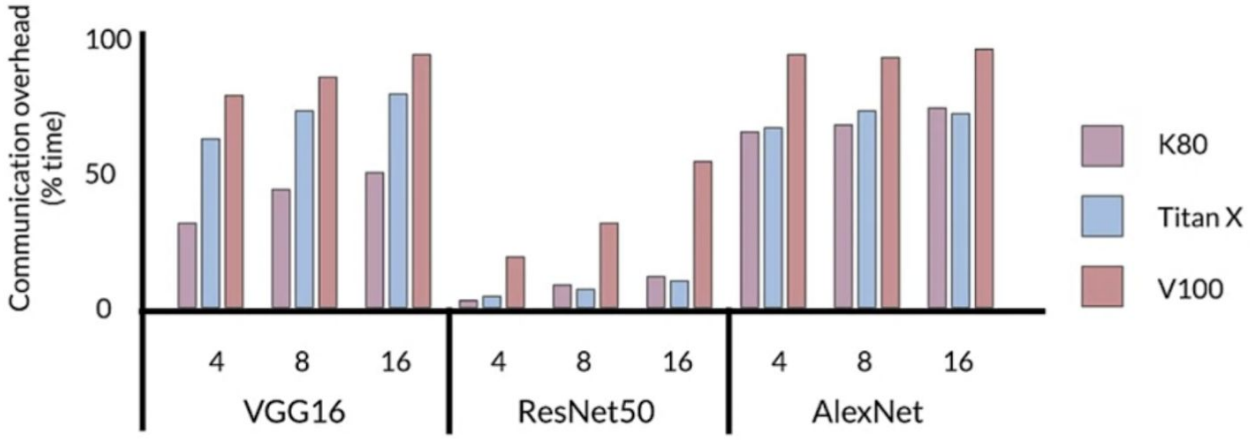
- N 개의 layer들을 k개의 그룹으로 나누어, 각 worker가 N/k layer들만큼 backprop.

More efficient:

- 각 layer마다 얼마나 compute intensive한지 분석해서 분배.

데이터의 병렬처리

Data parallelism: 하나의 모델이 여러 replica들로 만들어져 여러개의 accelerator(GPU/TPU)로 보내져 학습. 트레이닝데이터는accelerator 갯수만큼 나뉘어져서 개별학습.



Accelerator Compute Efficiency

- GPU/TPU 들에는 제한된 메모리만 존재.

Model parallelism: 대형 네트워크를 학습할 수는 있으나, accelerator compute의 효율도가 낮음.

Data parallelism: 다른 학습데이터로 병렬학습 가능하나, accelerator가 대형네트워크를 실행할 수 없음.

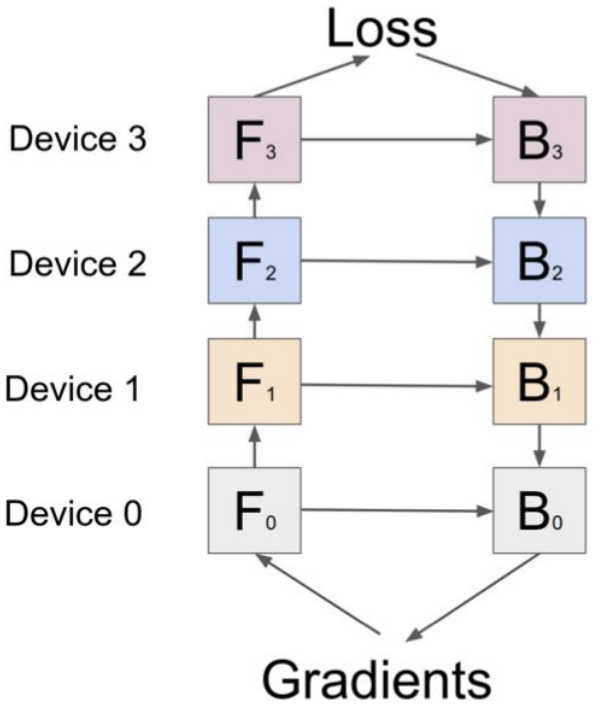
➔ Pipeline parallelism!

Pipeline Parallelism

Data parallelism과 Model parallelism을 혼합.

- (Data) Mini batch data를 micro-batch로 분할.
  - 각 다른 worker들은 다른 micro-batch data에 진행.
- (Model) 모델의 layer들을 partitioning

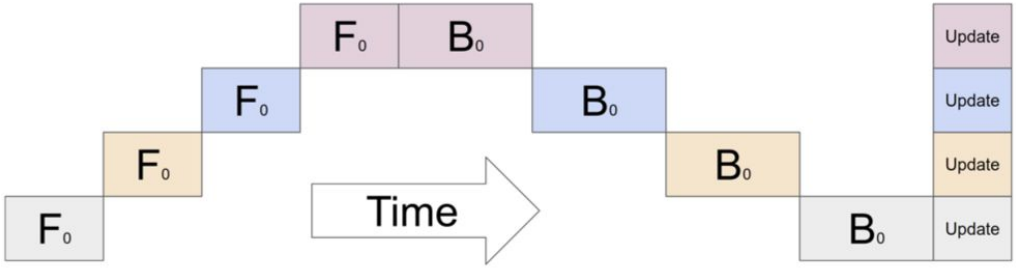
Pipeline Parallelism



4개의 accelerator에서 모델병렬처리:

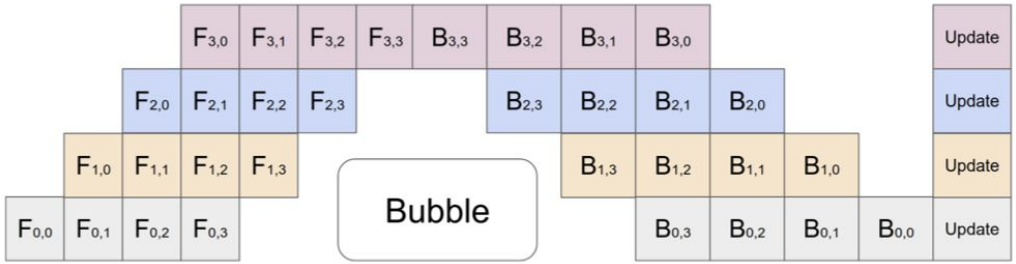
- F_i = i번째 accelerator에서 실행되는 forward pass
- B_i = i번째 acclerator에서 실행되는 backward pass
  - F_i와 B_(i-1)에 인풋으로 계산

Pipeline Parallelism



Naive:

- 네트웍의 순차적인 의존도때문에 낭비되는 idle time이 많다.



Parallelism:

- 각 batch를 micro-batch로 나눈 후, accelerator들이 동시에 micro-batch들을 처리 가능.

GPipe Algorithm

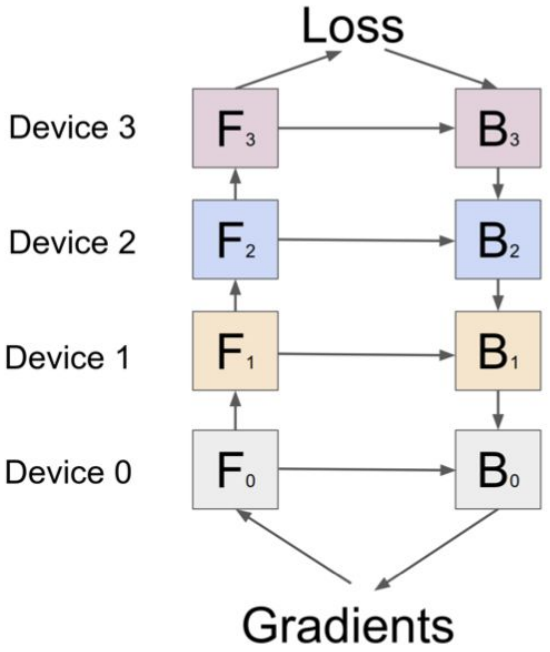
L_i = 모델에 존재하는 i번째 layer

f_i = forward computational function

w_i = f_i에 필요로 하는 weight parameter들.

세가지 parameter로 parallelism을 정의:

- K = 모델을 k개의 파티션으로 분할.
  - P_k = k 번째 모델 subgraph layers (layer L_i, L_i+1, ... L_j)
  - F_k = P_k 에서 실행하는 forward pass
- M = 분할/병렬하고자 하는 microbatch의 갯수
- L = 모델 레이어들.



GPipe Algorithm

1. 데이터셋중 배치를 M개의 micro-batch로 분할(파티션).
2. 이 데이터가 K개의 accelerator로 파이프라인 됨 (Forward pass)
3. Backward pass중에는:
   a. 각 micro-batch당 Gradient 을 계산
   b. Gradient 들이 누적되어 모델 파라메터가 K개에 전부 업데이트 됨.

파티셔닝 알고리즘= 각 cell마다의 estimated cost를 균일하게끔 분할.

각 파티션 바운더리마다 Communication 노드들이 자동으로 삽입 됨.

GPipe

Worker간의 통신:

- Partition boundary에 communication primitive가 들어가, batch간의 데이터/파라메터 통신 가능.
- Micro-batch간에 Gradient accumulation을 통해 모델 정확도를 유지.

System Optimization:

- Automatic parallelism을 통해 메모리 사용을 최적화.

GPipe

- Lingvo 프레임웍 위에 구현:
  - Lingvo: A TensorFlow Framework for Sequence Modeling (Jonathan Shen)
  - 시퀀스 모델에 적합한 프레임워크; 다른 프레임웍에도 gpipe 적용가능.
- Open source로 릴리즈: github
- 카카오에서 pytorch용 gpipe 릴리즈: torchpipe

Reference: *GPipe: Efficient Training of Giant Neural Networks using Pipeline Parallelism*

NVIDIA GPUs (8GB each)	Naive-1	Pipeline-1	Pipeline-2	Pipeline-4	Pipeline-8
AmoebaNet-D (L, D)	(18, 208)	(18, 416)	(18, 544)	(36, 544)	(72, 512)
# of Model Parameters	82M	318M	542M	1.05B	1.8B
Total Model Parameter Memory	1.05GB	3.8GB	6.45GB	12.53GB	24.62GB
Peak Activation Memory	6.26GB	3.46GB	8.11GB	15.21GB	26.24GB
Cloud TPUv3 (16GB each)	**Naive-1**	**Pipeline-1**	**Pipeline-8**	**Pipeline-32**	**Pipeline-128**
Transformer-L	3	13	103	415	1663
# of Model Parameters	282.2M	785.8M	5.3B	21.0B	83.9B
Total Model Parameter Memory	11.7G	8.8G	59.5G	235.1G	937.9G
Peak Activation Memory	3.15G	6.4G	50.9G	199.9G	796.1G

GPipe가 허용하는 AmoebaNet의 최대 모델 사이즈.

Pipeline-k = k개의 accelerator에 파티셔닝.

AmoebaNet-D(L, D) = L normal cell layers, 필터 사이즈 D

Transformer-L = L layer transformer

TPU	AmoebaNet			Transformer		
$K =$	2	4	8	2	4	8
$M = 1$	1	1.13	1.38	1	1.07	1.3
$M = 4$	1.07	1.26	1.72	1.7	3.2	4.8
$M = 32$	1.21	1.84	3.48	1.8	3.4	6.3

Training Throughput:

K= 파티션 갯수

M = microbatch 갯수

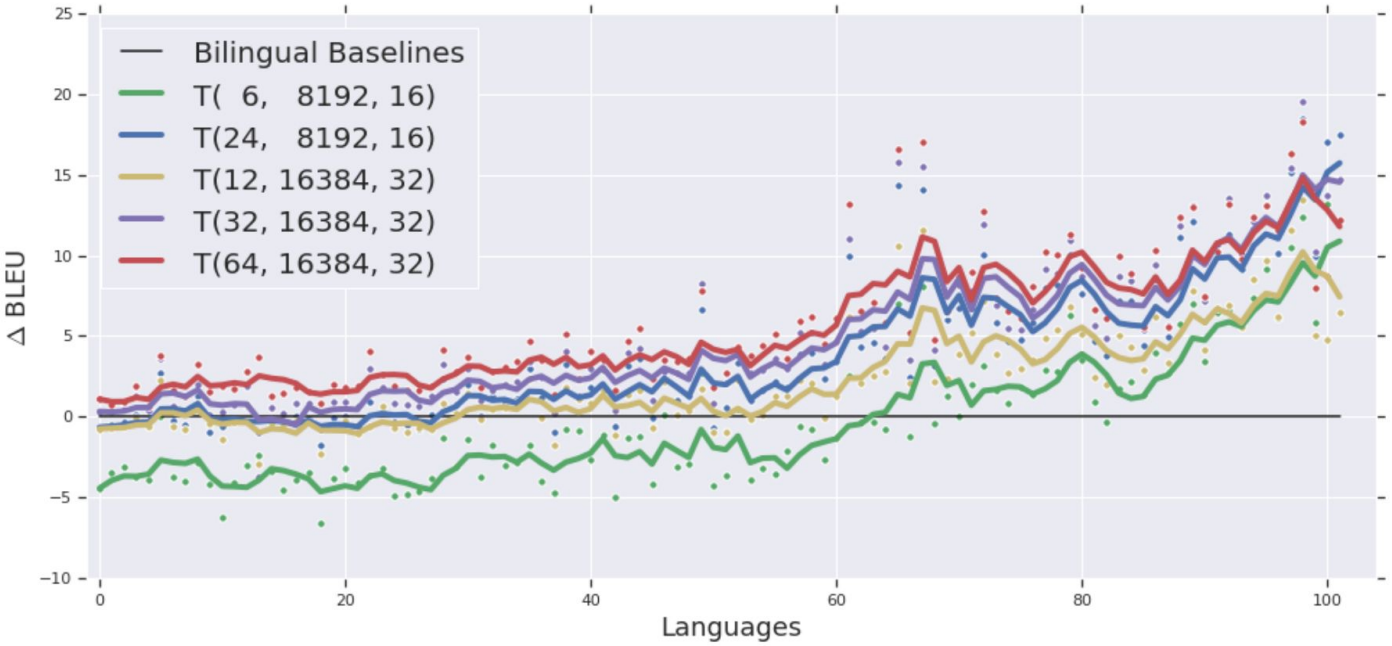
M>>K 인 Transformer의 경우, accelerator 갯수와 speedup이 거의 linear하게 향상!

Model Quality

Dataset	# Train	# Test	# Classes	Accuracy (%)	Previous Best (%)
ImageNet-2012	1,281,167	50,000	1000	**84.4**	83.9 [12] (85.4* [27])
CIFAR-10	50,000	10,000	10	**99.0**	98.5 [26]
CIFAR-100	50,000	10,000	100	**91.3**	89.3 [26]
Stanford Cars	8,144	8,041	196	94.6	**94.8*** [26]
Oxford Pets	3,680	3,369	37	**95.9**	93.8* [29]
Food-101	75,750	25,250	101	**93.0**	90.4* [30]
FGVC Aircraft	6,667	3,333	100	92.7	**92.9*** [31]
Birdsnap	47,386	2,443	500	**83.6**	80.2* [32]

Parallel training을 통해 large model기반으로 모델의 정확도도 향상.

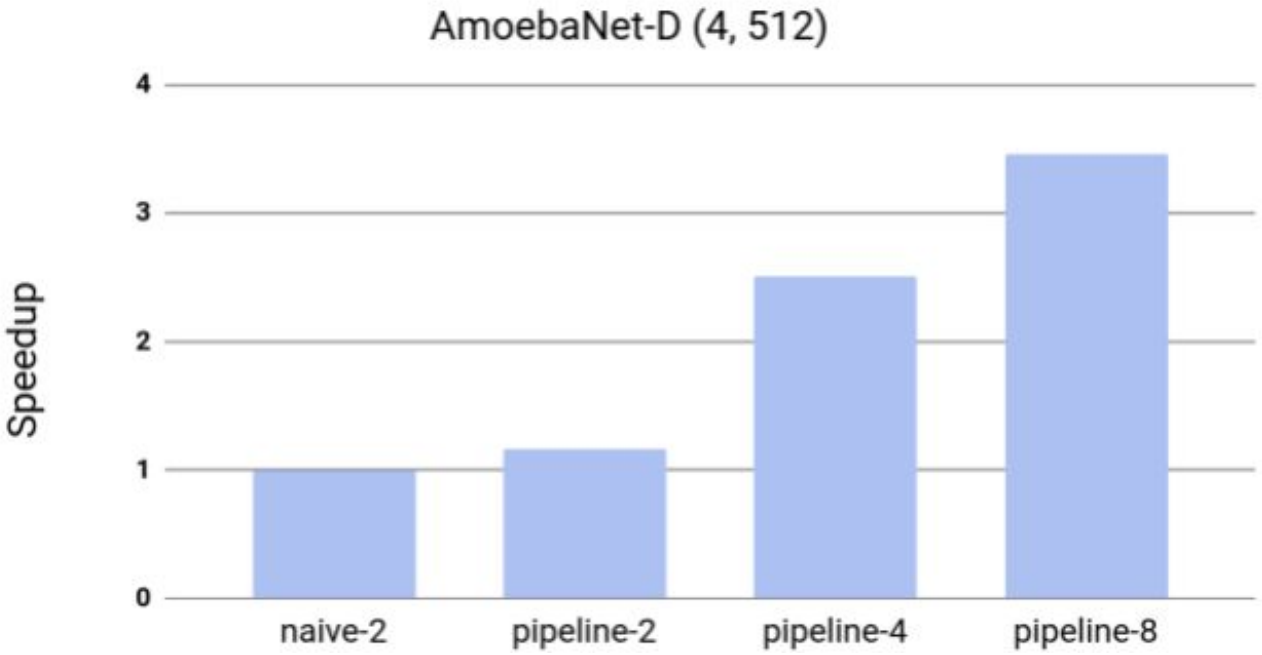
Model Quality



109 languages:

- 10k~1B train sample
- 400M~6B parameter

GPipe Benchmark



Reference

- Distributed training
- Data parallelism
- Pipeline parallelism
- GPipe
  a. Google AI Blog
  b. Pathways / PaLM 등의 기초마련

Break

중간고사 결과

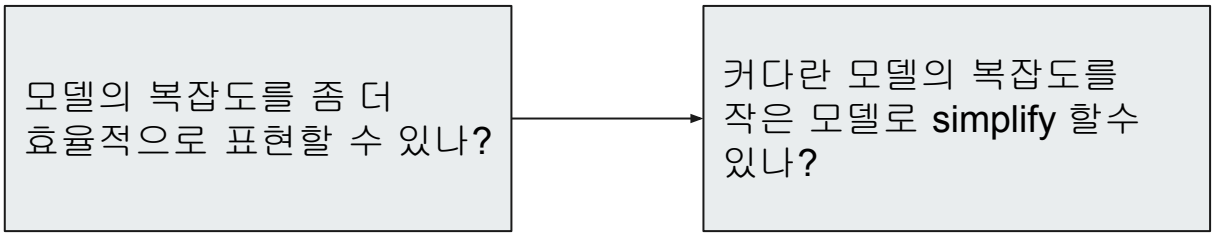
문제 3, 17, 18, 20번 리뷰.

답안지

Distillation

Large model의 이슈

- Large model일 수록, 모델의 복잡도가 증가한다.
- Task가 complex 할 수록, 모델의 복잡도가 필요할때가 종종 있다.



Knowledge Distillation의 필요성

Abstract

Deploy models

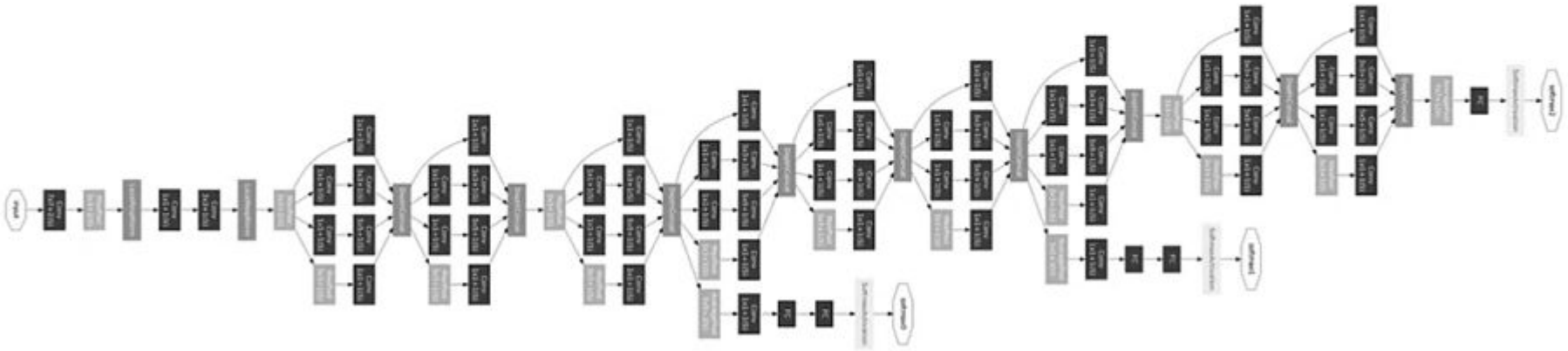
Q) But Could we use Ensemble when we deploy models?

A) Unfortunately, cumbersome & too computationally expensive

- to a large number of users
- the individual models are large NNs

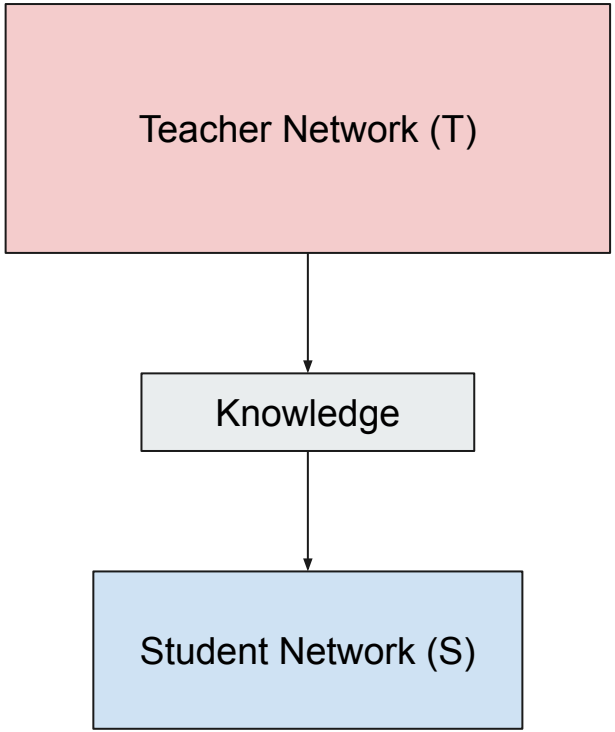
복잡한 ensembl 모델을 디플로이 한다고 가정했을 때, edge device처럼 컴퓨팅 리소스가 낮은 경우 복잡한 모델은 부적합.

예: GoogLeNet



Google Inception (GoogLeNet): 2014 ImageNet winner

Knowledge Distillation



Teacher Network (T)

- Ensemble / large model
- (+) High model accuracy
- (-) train / serving 둘다 compute heavy

Student Network (S)

- Small model
- (+) 여러 환경에 deploy하기에 적합
- (-) low model accuracy than T.

*Hinton; Vinyals; Dean (**2015**). "Distilling the knowledge in a neural network".*

Distillation with Soft Label

classification모델의 경우 Hard label을 이렇게 표현할 수 있음.

cow	dog	cat	car
0	1	0	0

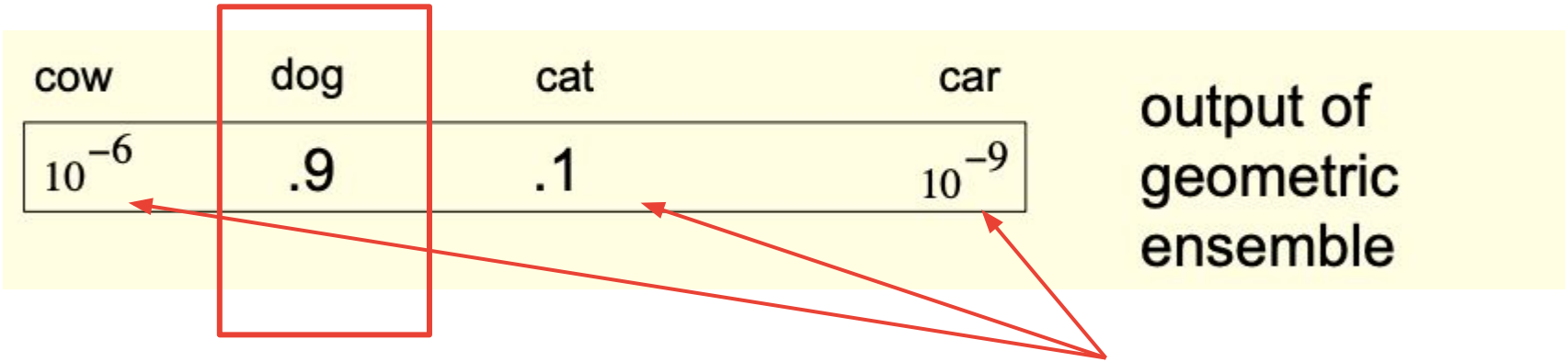
original hard targets

이를 마지막 softmax layer를 통해 각 클래스당 확률값을 계산.

$$q_i = \frac{exp(z_i)}{\sum_j exp(z_j)}$$

Distillation with Soft Label

Softmax layer를 써서 계산한 값중 가장 높은 값을 predicted class로 아웃풋을 함.



여기서 나머지 class들이 상대적으로 얼마나 높은 점수를 받았는지를 student model에게 transfer하고 싶다. ("Dark Knowledge")

그러나 너무 dog에게 score가 집중되어 나머지 값들의 비교가 어려움 -> 좀더 "soft"한 점수 배포가 필요함.

Distillation with Soft Label

$$q_i = \frac{exp(z_i/T)}{\sum_j exp(z_j/T)}$$

“Softened output”

T: “temperature”

- T= 1일 때는 hard output
- T>>> 1일 때는 softened output

cow	dog	cat	car
.05	.3	.2	.005

softened output of ensemble

Teacher와 Student의 학습목적

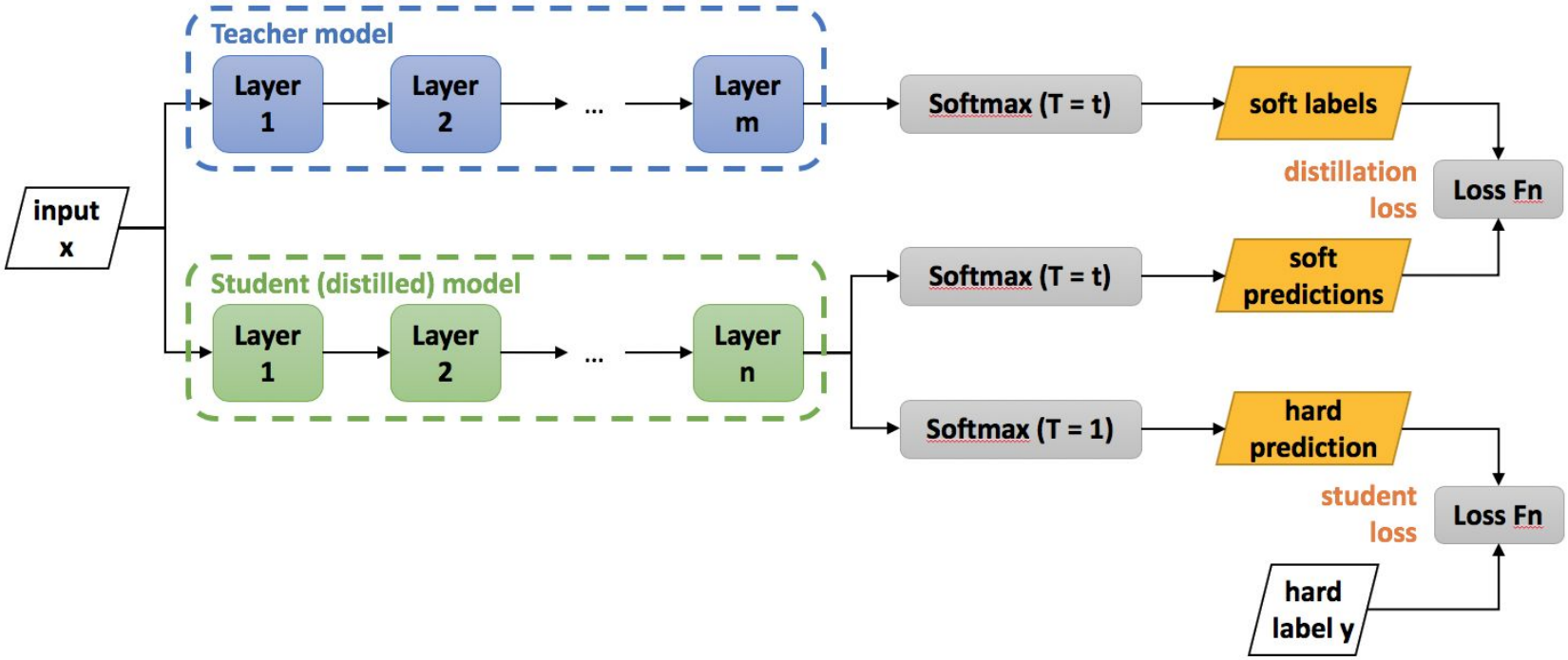
Teacher:

- 정상적인 학습
- 원래 문제의 metric으로 학습

Student:

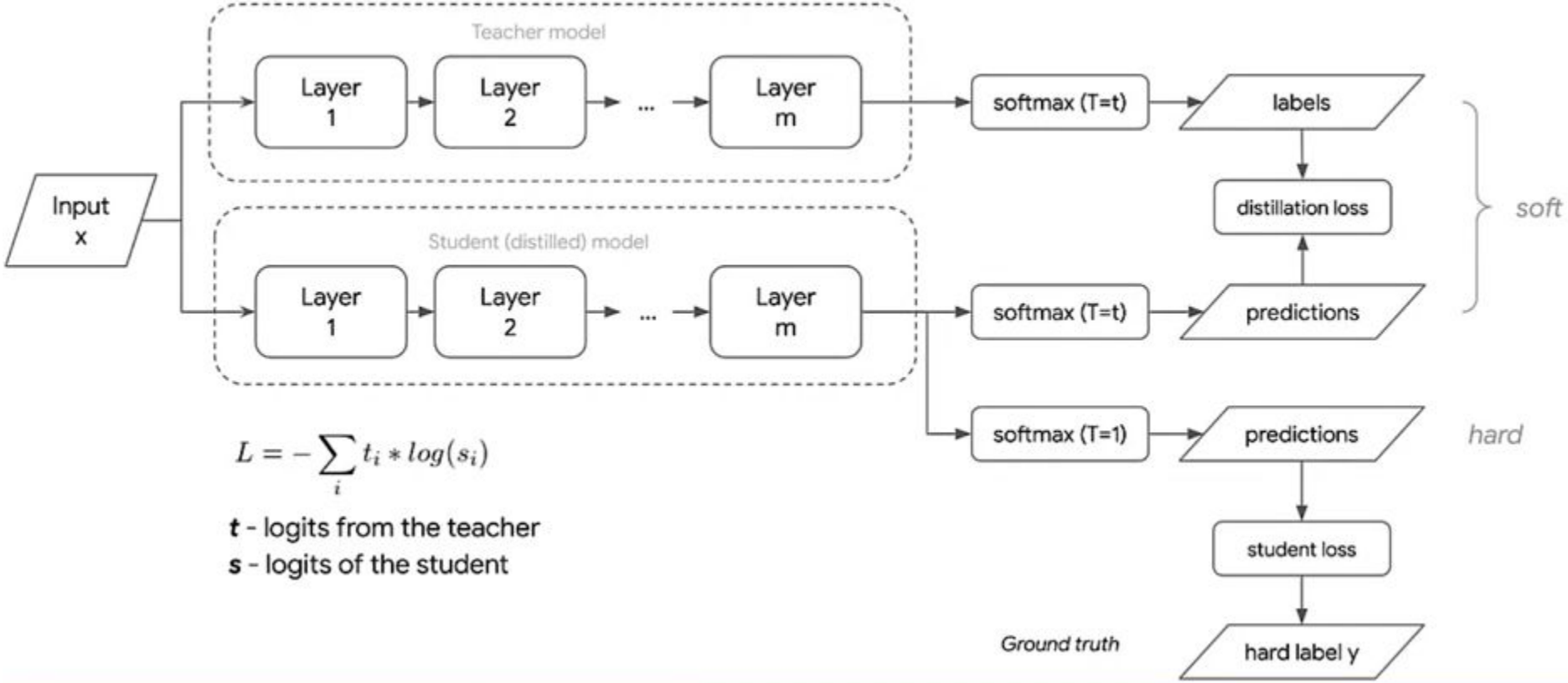
- T로부터 Knowledge transfer가 목적
- Teacher의 output probability distribution을 통해 soft target을 매칭.
    - “Soft target”를 T의 knowledge 를 학습.

Knowledge Transfer



$$\mathcal{L}(x; W) = \alpha * \mathcal{H}(y, \sigma(z_s; T = 1)) + \beta * \mathcal{H}(\sigma(z_t; T = \tau), \sigma(z_s, T = \tau))$$

How knowledge transfer takes place



$$L = -\sum_i t_i * log(s_i)$$

t - logits from the teacher

s - logits of the student

Model Accuracy

Model	Accuracy	Word Error Rate (WER)
Baseline	58.9%	10.9%
10x Ensemble	61.1%	10.7%
Distilled Single Model	60.8%	10.7%

ASR model (Android)

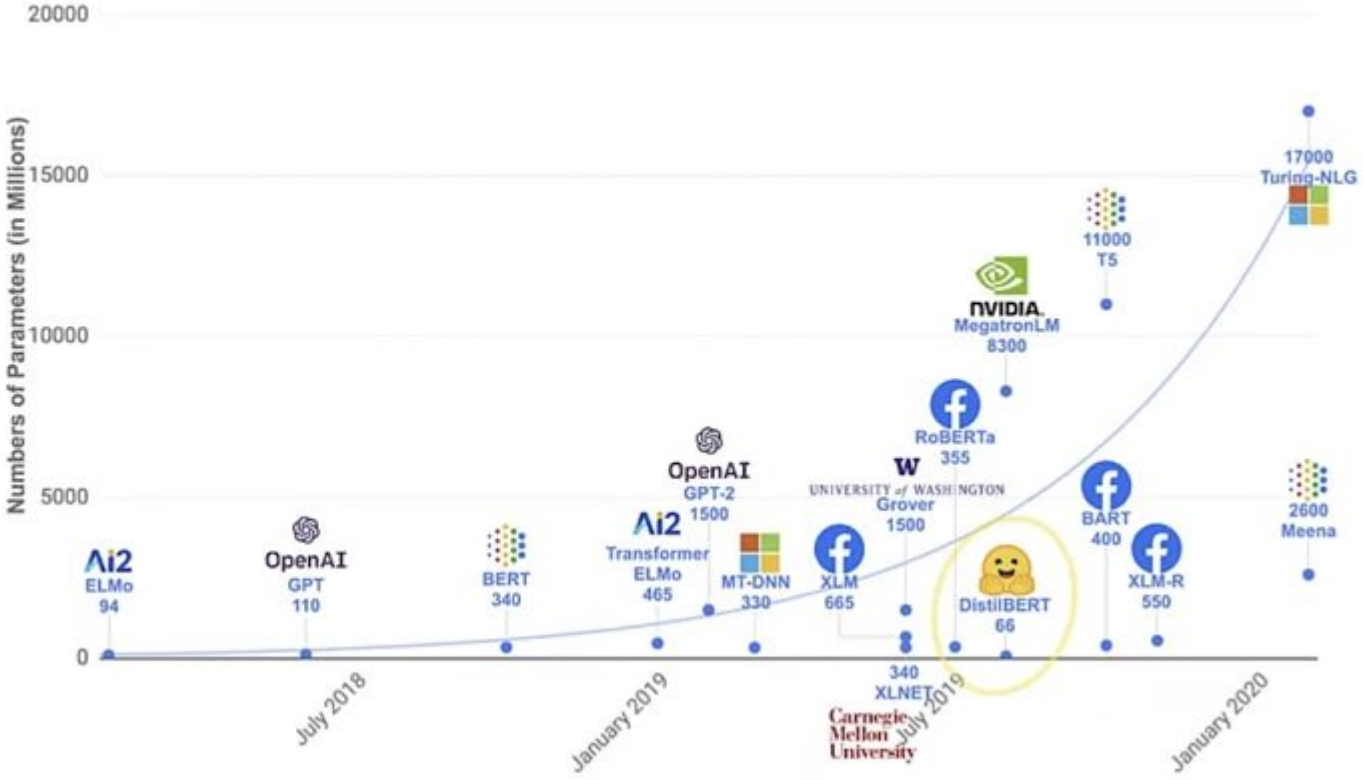
Baseline: 8 hidden layers, each with 2560 relu + final softmax with 14000 labels.

10xEnsemble: 10 separate models ensembled together

Distill: single student model. **Almost as good as ensemble!**

DistilBERT

40% fewer params,
runs 60% faster while
preserving 97% of
model quality (GLUE)



Colab

Knowledge Distillation Colab